Resilient, Communication-Reducing, and Adaptive Time Stepping to Accelerate Exascale Scientific Applications

Archibald[1], Constantinescu[2], Evans[1], Finkel[2], Haut[3], Norris[2], Norman[1], Sandu[4], Stoyanov[1], Tokman[5], Wingate[3], and Xing[1]

[1]Oak Ridge National Laboratory, TN 37831 ({archibaldrk,evanskj,normanmr,stoyanovmk,xingy}@ornl.gov)
[2]Argonne National Laboratory, IL 60439 ({emconsta,hfinkel,norris}@anl.gov)
[3]LANL, NM 87545 ({terryhaut,wingate}@lanl.gov)
[4]Virginia Tech, VA 24060 (sandu@cs.vt.edu)
[5]UC Merced, CA 95340 (mtokman@ucmerced.edu)

Summary of Exascale Position

Many applications of interest to DOE involve very large simulations of systems modeled by time dependent partial differential equations. Current time discretization methods face considerable challenges at extreme scales: the time dimension does not parallelize well; global time steps are driven by only a few components; traditional implicit discretizations, and the synchronization at the end of each time step, create global dependencies; and failure of one node stalls the entire simulation.

At the same time, the nature of time-stepping methods presents unique and untapped mathematical opportunities for achieving high end scalability and resiliency. Detection and correction of errors across the system for various high performance computing architectures can be handled algorithmically within the framework of various long time-stepping methods with minimal overhead due to existing parity of information, namely, the previous, current, and intermediate (stage) simulation states.

Highly Scalable Time Discretizations

Traditional time stepping methods advance the entire simulation using a serial global-time step. This leads to serious limitations of parallelism at the exascale. The size of the global time steps is driven by the fastest varying components of the system, which can be unnecessarily small for the slower varying components, therefore wasting computational resources. Second, a global barrier-like synchronization of the entire simulation is required at the end of each (macro)step/stage, which may lead to considerable data dependencies among different tasks, as well as to severe load imbalances. Moreover at the exascale latency and energy demand becoming a major limiting factors and concerns. Fully-implicit (FI) methods have demonstrated the ability to converge multi-physics, multi-scale nonlinear application at current computational scale [15, 16] and will belong to suite of time stepping methods that must be advance in order to prepare for the exascale.

Parallel in time methods can take advantage of the computational resources to perform independent calculations in order to enhance the robustness of the integration algorithm by improving the resiliency [13] stability and accuracy [12, 40, 44]. Currently, parallelism in the time-stepping algorithms at the time discretization level has been mainly explored in the context of multistep [50] and multistage methods [6, 17, 24, 25, 49, 51]. However, resiliency has not been addressed in the context of the studies mentioned and implementation is limited [25], several research activities must be conducted to expose more levels of parallelism while limiting communication in order to reach the exascale.

Asynchronous multirate time stepping takes different time steps for different components to achieve a global target accuracy. Early efforts to develop multirate RK methods are due to Rice [42] and Andrus [1, 2]. Multirate versions of many of the traditional time stepping schemes have been proposed, including linear multistep [18, 27], extrapolation [12, 44], RK [19, 28, 29], Rosenbrock-Wanner [4, 20], waveform relaxation [31, 43], Galerkin [33–35], and combined multiscale [11] ap-

proaches. In addition, the multirate approach can be extended to exponential integrators which have been recently developed as efficient alternatives to explicit and implicit methods for large stiff systems of differential equations [21, 22, 32, 47, 48]. Building asynchronicity into multirate timesteping will produce additional levels of parallelism and provide insights on how to limit communication and build in algorithmic resiliency.

Furthermore, Arbitrary DErivative Riemann (ADER) temporal discretizations provide an accurate, flexible, and efficient framework for high-order-accurate, single-stage, explicit time-stepping for the solution of PDEs. Single-stage methods are advantageous in massively parallel environments by clustering computation and reducing data transfer frequency compared to multi-stage methods. The ADER method [46] uses the definition of the PDE and provided spatial derivatives to construct space-time derivatives to any desired order. These derivatives can be computed efficiently using Differential Transforms (DTs) [38, 39]. Computing space-time derivatives only once per cell, saving DTs of flux and source terms, and expanding space-time Taylor series of each PDE term mean sampling and averaging are always performed on polynomials. This can be done analytically, removing the need for quadrature. Polynomial evaluation and integration as well as DT computations contain fine-grained, data-parallel structures that may be exploited on accelerated architectures such as GPUs. Finally, ADER methods using DTs are very easily adapted to any order of accuracy, any spatial operator, and any mesh. In conjunction with a multi-moment finite-volume operator, ADER efficiently operates at extremely large time steps, epitomizing the current push toward communication avoidance in parallel.

Algorithmic Fault Tolerance and Resiliency

Exascale computing systems are expected to have processor cores, memory units, communications and other components totaling in the numbers of millions [10], and computations that run on these systems for a few hours are likely to experience failures of several components, and possibly composite failures that cannot always be predicted and accounted for in advance. Methods for achieving robust computations using failure-prone computing systems have been developed in several cases [41]. Von Neumann [52] studied the mathematical aspects of building reliable computing systems from unreliable components in the 50s, and related works establish that robust computations can be achieved by using inherently failure-prone systems. In practice, computations in space vehicles are enhanced with Software-Implemented Hardware Fault Tolerance (SIHFT) methods to counteract the transient faults due to radiation exposure [14]. These studies show that it is indeed possible to achieve robust computations in production systems in some of the most challenging, failure-prone environments. A wide spectrum of analytical methods and deployed systems exist [3, 7], ranging from processor design [45], to programs that check their work [5], to methods specific to MPI [9, 36, 54], to process migration [53] (to name a few).

In addition to utilizing designs mentioned above, developing algorithmically resilient time integrators in an exascale environment will require run time performance measures to adapt computational structures, such as time windows subdomains etc. to the current state of the computational system. A simulation model of the given architecture can be used to emulate performance effects of work and data distribution policies on proposed time stepping algorithms on future architectures. A variety of tools is readily available (e.g., [8, 23, 26, 37]), consisting of a combination of discrete event simulators (DES) for the performance of each node and the communication times as well as linear programming tools for optimization. However this modeling for the exascale will be expensive to run but even more expensive to maintain. Hence, there has been a large research effort in recent years to develop scalable aggregate, continuous descriptions of these DES (a review is given in [30]). Runtime performance measurements on new architectures are needed to manage faulty environments.

[1] J.F. Andrus. Numerical solution for ordinary differential equations separated into subsystems. *SIAM Journal on Numerical Analysis*, 16(4):605–611, 1979.

[2] J.F. Andrus. Stability of a multi-rate method for numerical integration of ODEs. *Computers & Mathematics with applications*, 25(2):3–14, 1993.

[3] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1:11–33, 2004.

[4] A. Bartel and M. Günther. A multirate W-method for electrical networks in state-space formulation. *Journal of Computational and Applied Mathematics*, 147(2):411–425, 2002.

[5] M. Blum and S. Kannan. Designing programs that check their work. *J. ACM*, 42(1):269–291, 1995.

[6] K. Burrage. Parallel methods for initial value problems. *Applied Numerical Mathematics*, 11(1):5–25, 1993.

[7] F. Cappello, A. Geist, B. Gropp, S. Kale, B. Kramer, and M. Snir. Towards exascale resilience. *Journal of High Performance Computing Applications*, 23(4):374–388, 2009.

[8] H. Casanova, A. Legrand, and M. Quinson. Simgrid: a generic framework for large-scale distributed experiments. In *10th IEEE International Conference on Computer Modeling and Simulation - EUROSIM / UKSIM*, Cambrige, United Kingdom, 2008.

[9] S. Chakravorty, C. L. Mendes, and L. V. Kale. Proactive fault tolerance in mpi applications via task migration. In *Proceedings of HIPC*, page 485, 2006. LNCS volume 4297.

[10] J. Dongarra, P Beckman, and et al. The international exascale software roadmap. *International Journal of High Performance Computer Applications*, 25(1), 2011.

[11] B. Engquist and R. Tsai. Heterogeneous multiscale methods for stiff ordinary differential equations. *Mathematics of Computation*, 74:1707–1742, 2005.

[12] C. Engstler and C. Lubich. Multirate extrapolation methods for differential equations with different time scales. *Computing*, 58(2):173–185, 1997.

[13] WH Enright and DJ Higham. Parallel defect control. *BIT Numerical Mathematics*, 31(4):647–663, 1991.

[14] P. P. Shirvani et al. Software-implemented hardware fault tolerance experiments: Cots in space. In *Proceeding of International Conference on Dependable Systems and Networks*, pages 56–57. Fast Abstracts, New York, NY, 2000.

[15] K. J. Evans, D. Rouson, A. G. Salinger, M. Taylor, W. Weijer, and J. B. White III. A scalable and adaptable solution framework within components of the community climate system model. *Lecture Notes in Comp. Sci.*, 5545:332–341, 2009.

[16] K. J. Evans, A. G. Salinger, P. H. Worley, S. Price, W. Lipscomb, J. Nichols, J. B. White III, M. Perego, J. Edwards, M. Vertenstein, and J.-F. Lemieux. A modern solver template to manage solution algorithms in the community earth system mode. *The International Journal of High Performance Computing Applications*, 26:54–62, 2012.

[17] CW Gear. Massive parallelism across space in odes. *Applied Numerical Mathematics*, 11(1-3):27–43, 1993.

[18] C.W. Gear and D.R. Wells. Multirate linear multistep methods. *BIT*, 24:484–502, 1984.

[19] M. Günther, A. Kværnø, and P. Rentrop. Multirate partitioned runge-kutta methods. *BIT Numerical Mathematics*, 41(3):504–514, 2001. 0006-3835.

[20] M. Günther and P. Rentrop. Multirate ROW-methods and latency of electric circuits. *Applied Numerical Mathematics*, 13(1-3):83–102, 1993.

[21] T. Haut and B. Wingate. An asymptotic parallel-in-time method for highly oscillatory pdes. *SIAM Journal of Scientic Computing*, Submitted, 2013.

[22] M. Hochbruck and A. Ostermann. Exponential integrators. *Acta Numerica*, pages 209–286, 2010.

[23] T. Hoefler, G. Bronevetsky, B. Barrett, B. R. de Supinski, and A. Lumsdaine. Efficient mpi support for advanced hybrid programming models. In *Recent Advances in the Message Passing Interface (EuroMPI'10)*, volume LNCS 6305, pages 50–61. Springer, 2010.

[24] K.R. Jackson. A survey of parallel numerical methods for initial value problems for ordinary differential equations. *IEEE Transactions on Magnetics*, 27(5):3792–3797, 1991.

[25] KR Jackson and S.P. Norsett. The potential for parallelism in Runge-Kutta methods. Part 1: RK formulas in standard form. *SIAM journal on numerical analysis*, pages 49–82, 1995.

[26] C. L. Janssen, H. Adalsteinsson, and J. P. Kenny. Using simulation to design extremescale applications and architectures: programming model exploration. *ACM SIGMETRICS Performance Evaluation Review*, 38:4–8, 2011.

[27] T. Kato and T. Kataoka. Circuit analysis by a new multirate method. *Electrical Engineering in Japan*, 126(4):55–62, 1999.

[28] A. Kværnø. Stability of multirate Runge-Kutta schemes. *International Journal of Differential Equations and Applications*, 1(1):97–105, 2000.

[29] A. Kværnø and P. Rentrop. Low order multirate runge-kutta methods in electric circuit simulation, 1999.

[30] E. Lefeber and D. Armbruster. Aggregate modeling of manufacturing systems. In K.G. Kempf, P. Keskinocak, and R. Uzsoy, editors, *Planning Production and Inventories in the Extended Enterprise: A State of the Art Handbook*, pages 509 – 536. Springer Verlag, 2010.

[31] E. Lelarasmee, A.E. Ruehli, and A.L. Sangiovanni-Vincentelli. The waveform relaxation method for time-domain analysis of large scale integrated circuits. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 1(3):131–145, 1982.

[32] J. Loffeld and M. Tokman. Comparative performance of exponential, implicit, and explicit integrators for stiff systems of odes. *The Journal of Computational and Applied Mathematics*, 241:45–67, 2013.

[33] A. Logg. Multi-adaptive Galerkin methods for ODEs I. *SIAM Journal on Scientific Computing*, 24(6):1879–1902, 2003.

[34] A. Logg. Multi-adaptive Galerkin methods for ODEs II: Implementation and applications. *SIAM Journal on Scientific Computing*, 25(4):1119–1141, 2003.

[35] A. Logg. Multi-adaptive Galerkin methods for ODEs III: A priory estimates. *SIAM Journal on Numerical Analysis*, 43(6):2624–2646, 2006.

[36] C. Lu and D. A. Reed. Assessing fault sensitivity in mpi applications. In *Proceedings of the 2004 ACM/IEEE conference on Supercomputing*, 2004.

[37] M.O. McCracken and A. Snavely. A simulation toolkit to investigate the effects of grid characteristics on workflow completion time. In *Proceedings of the 4th Workshop on Workflows in Support of Large-Scale Science*, 2009.

[38] M.R. Norman. Algorithmic improvements for schemes using the {ADER} time discretization. *Journal of Computational Physics*, 243(0):176 – 178, 2013.

[39] M.R. Norman and H. Finkel. Multi-moment ADER-Taylor methods for systems of conservation laws with source terms in one dimension. *Journal of Computational Physics*, 2012.

[40] T. Rauber and G. Rünger. Load balancing schemes for extrapolation methods. *Concurrency: Practice and Experience*, 9(3):181–202, 1997.

[41] D. A. Rennels. Fault-tolerant computing. In E. Reilly A. Ralston and D. Hemmendinger, editors, *Encyclopedia of Computer Science*. International Thomson Publishing, 1999.

[42] J.R. Rice. Split Runge-Kutta methods for simultaneous equations. *Journal of Research of the National Institute of Standards and Technology*, 64(B):151–170, 1960.

[43] J. Sand and K. Burrage. A Jacobi waveform relaxation method for ODEs. *SIAM Journal on Scientific Computing*, 20(2):534–552, 1998.

[44] Adrian Sandu and Emil Constantinescu. Multirate explicit adams methods for time integration of conservation laws. *Journal of Scientific Computing*, 38:229–249, 2009. 10.1007/s10915-008-9235-3.

[45] R. D. Schlichting and F. B. Schneider. Fail-stop processors: an approach to designing fault-tolerant computing systems. *ACM Transactions on Computer Systems*, 1(3), 1983.

[46] V. A. Titarev and E. F. Toro. ADER: Arbitrary High Order Godunov Approach. *Journal of Scientific Computing*, 17(1):609–618, December 2002.

[47] M. Tokman. A new class of exponential propagation iterative methods of Runge-Kutta type (EPIRK). *Journal of Computational Physics*, 230:8762–8778, 2011.

[48] M. Tokman, J. Loffeld, and P. Tranquilli. New adaptive exponential propagation iterative methods of runge-kutta type. *SIAM Journal on Scientific Computing*, 34:A2650–A2669, 2012.

[49] P.J. van der Houwen and N.H. Cong. Parallel block predictor-corrector methods of Runge-Kutta type. *Applied Numerical Mathematics*, 13(1-3):109–123, 1993.

[50] P.J. van der Houwen and E. Messina. Parallel Adams methods. *Journal of computational and applied mathematics*, 101(1):153–165, 1999.

[51] PJ van der Houwen and BP Sommeijer. Analysis of parallel diagonally implicit iteration of Runge-Kutta methods. *Applied Numerical Mathematics*, 11(1):169–188, 1993.

[52] J. von Neumann. Probabilistic logics and the synthesis of reliable organisms from unreliable components. In C. E. Shannon and J. MaCarthy, editors, *Automata Studies*. Princeton University Press, 1956.

[53] C. Wang, F. Mueller, C. Engelmann, and S. L. Scott. Proactive process-level live migration in hpc environments. In *Proceeding of Supercomputing*, 2008.

[54] G. Zheng, L. Shi, and L. V. Kale. FTC-charm++: An in-memory checkpoint-based fault tolerant runtime for Charm++ and MPI. In *IEEE International Conference on Cluster Computing*, pages 93–103, 2004.